Manipulation des layouts

version 1

Interface Homme-Machine : Unity

Voici les objectifs de ce sujet :

— Continuez à manipuler l'IDE `Unity`.

— Continuez la création d'un *widget* complexe.

— Exploitez les mécaniques vues précédemment.

— L'utilisation des Layouts

Description générale du TP

La fois précédente, nous avons réalisé nos premiers widget complexe en exploitant l'agrégation de plusieurs widget de base. Nous avons en particulier manipuler le système d'ancre que propose Unity pour placer les objets de façon relative.

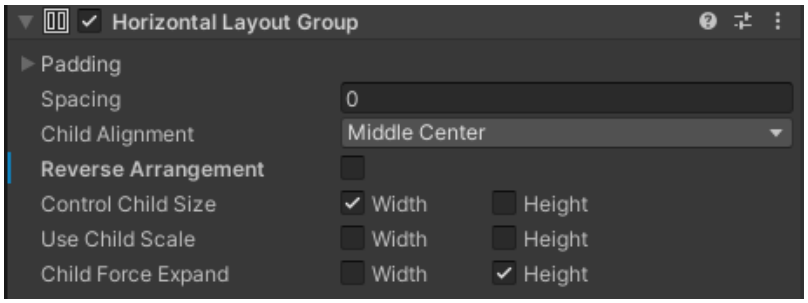
Ici, nous allons voir une autre méthode un peu plus couteuse mais offrant une plus grande puissance en terme de placement et qui reprend les points que vous avez étudié dans les années précédentes en IHM : les mises en pages (layout). La documentation est présente ici : `https://docs.unity3d.com/Packages/com.unity.ugui@1.0/manual/UIAutoLayout.html`

Présentation des outils de mise en pages en Unity UI

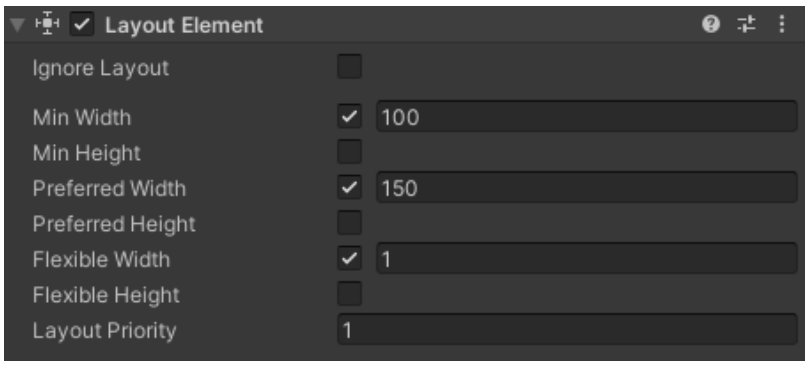
Il est conseillé de lire en détail la documentation cité plus haut pour bien comprendre les aspects et tous les détails. Je me contente ici de résumer des points clés :

— Vous avez 2 types de composants dédiés à la mise en page

— Les conteneurs ou *Layout Group* contrôlent le comportement des widgets fils (enchainement horizontal, enchainement vertical ou sous forme de grille, ...)



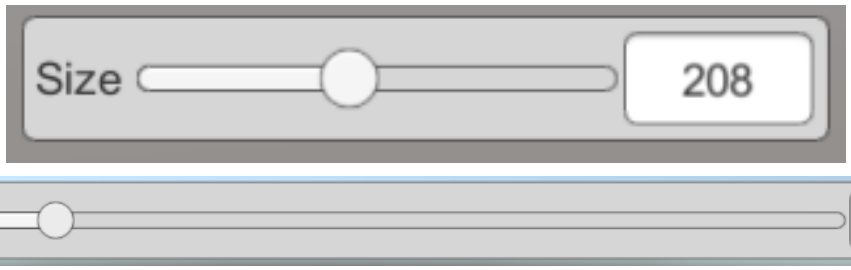
— Les composants élément ou *Layout Element* qui indique leur présence de mise en page.



Ainsi, chaque conteneur peut avoir des paramétrages différents qui vont faire évoluer les éléments selon les contraintes. Vous ferez attention à certains paramètres qui peuvent forcer le redimensionnement des widgets éléments sans les consulter même si leur taille n'était pas voulue. Vous ferez aussi attention aux éléments qui doivent activer la flexibilité des dimensions voulues pour agrandir dans cette direction (une valeur de 1 peut être suffisant pour indiquer un degré de liberté).

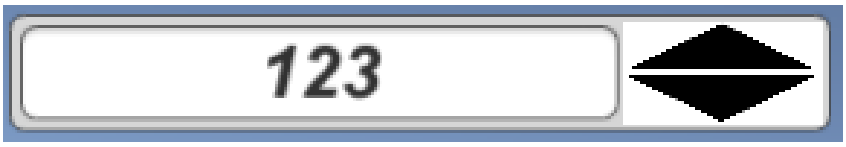
1 Widget : ComplexSlider le retour en joli

Refaite le widget ComplexSlider (soit sous un autre nom, soit après avoir fait une sauvegarde de votre ancien projet/-widget). Pour obtenir un affichage joli peu importe la largeur que vous donnerez à votre widget pour que le slider prenne la plus grande place.



2 Widget : Spinner

Réalisez le widget du Spinner qui consiste à contrôler les évolutions d'un nombre via 2 boutons regroupés en bout de ligne.



Le widget doit être fonctionnel, mais vous pouvez bien sur changer/adapter selon vos souhaits le coté esthétique du widget.

3 Aspects avancés sur le système d'événement souris

Le système événementiel est en cours de modification, mais nous présentons ici une mécanique pour interagir avec des événements particuliers. Pour cela, je vous renvoie sur le lien suivant qui donne les événements supportés `https://docs.unity3d.com/Packages/com.unity.ugui@1.0/manual/SupportedEvents.html`. Pour cela, nous vous proposons un petit exercice sous forme de tutoriel :

- Dans un nouveau projet, ajoutez un panel occupant l'entièreté du Canvas.
- Modifiez la couleur du panel pour qu'il soit entièrement transparent.
- Ajoutez un nouveau script.
- Au début du fichier script, ajoutez la ligne `using UnityEngine.EventSystems;`.
- Faire hériter notre script avec les événements voulues (cf. lien plus haut). Dans notre exemple nous allons nous concentrer sur les cliques souris et donc nous prendrons l'interface : `IPointerClickHandler`.
- Surchargez les fonctions associées aux interfaces. Ici : `public void OnPointerClick(PointerEventData data) { }`

Normalement, les événements fait avec cette mécanique réagisse normalement et vous pouvez le vérifier avec des messages de Log.

A présent, nous souhaitons réalisez les points suivants : lorsque nous cliquons sur une zone de l'écran, nous voulons créer à la volée un widget de notre choix à l'écran en tant que fils de notre Panel initial (n'oubliez pas un widget UI doit avoir comme parent un Canvas!!!). Pour réalisez cela, nous regarderons la fonction `https://docs.unity3d.com/ScriptReference/Object.Instantiate.html`.

Réalisez un exemple simple et que remarquez vous sur le pivot de vos ajouts ?

Enfin, vous êtes prêt à créer un script `ResizeWidget` qui consiste à agrandir en largeur/hauteur un widget si nous faisons un drag sur un widget.